

wherein dynamic information comprises statuses of said plurality of processing modules and modifications to said instantiated scripts including processed data from previously processed instructions, and wherein during execution of said instantiated scripts, said task module provides said dynamic information to said instantiated scripts and incorporates said dynamic information into said currently processing instructions for real-time consideration thereof, and upon completion of said currently processing instructions, said task module evaluates said incorporated dynamic information and processed data from said completed instructions and selectively executes said instantiated scripts such that instantiated scripts proceed to a second of said processing modules for processing of a next instruction within said instantiated scripts.

REMARKS

Applicants' representatives thank Examiner Susan Lao and SPE Alvin Oberley for the courtesy of a number of in-person and telephonic interviews previously granted during prosecution of this application. While such courtesy is appreciated, it is submitted that the Applicants have undergone substantial hardship due to the prolonged examination of this application. A review of the file history of this application reveals five (5) Office Actions, many of which included new grounds for rejecting the pending claims.

In this Amendment and Reply, Applicants have fully and completely addressed the Examiner's new grounds for rejection. It is hoped that further search by the Examiner is unnecessary and that this application can immediately proceed to allowance.

In the present Office Action claims 1-46 and 60-63 were examined. Claims 1-46 and 60-63 were rejected, and no claims were allowed.

By this Amendment and Response, clarifying amendments were made to independent claims 1, 33, 34, 60 and 63, no claims were cancelled or added. Accordingly, claims 1-46 and 60-63 are still pending in the application and, as now written, are believed to be in a condition for allowance.

Support for the amendments to independent claims 1, 33, 34, 60 and 63 may be found in the application and drawings as filed, and at least at page 13, lines 3-9, page 35, line 15 to page 38, line 12, and FIG. 4. Thus, no new matter was added.

Reconsideration and allowance in view of the amendments and remarks to follow is respectfully requested.

Prior Art Rejections:

Claims 1-46, 60-61 and 63 are rejected under 35 U.S.C. 103(a) as being unpatentable over Skillman et al. (U.S. Patent No. 5,506,999) and Fischer (U.S. Patent No. 5,337,360). Additionally, claim 62 is rejected under 35 U.S.C. 103(a) as being unpatentable over Skillman et al. and Fischer as applied to claim 60 and further in view of Waclawsky et al. (U.S. Patent No. 5,493,689). These rejections are respectfully disagreed with and are traversed below.

The cited documents are merely seen to describe a static execution and/or interpretation of a sequential set of instructions by one or more processing module. Applicants submit that the present invention teaches and, as presently written, claims an evaluation of dynamic information received during execution of an instantiated script and incorporated for real-time evaluation of such information within a currently processing instruction and, based thereon, to direct execution of a next instruction by a next processing module.

For example, and with respect to claim 1, the Examiner states, in summary, that Skillman et al. describes substantially all the limitations recited in claim 1, as previously written, except that "Skillman does not explicitly teach that the instructions/script proceed from a first one to a second one of the distributed processing modules for processing a next instruction of the overall processing." The Examiner replies on Fischer to cure this deficiency and states that "Fischer teaches distributed data processing, wherein instructions (traveling program) proceed from a first one to a second one of the distributed processing modules (transmits itself to the next destination) for processing a next instruction in the sequence of the overall processing (to collect, edit and approve data)." The characterization of these documents and the proposed combination thereof is respectfully disagreed with, and traversed below.

Skillman et al. describe a blackboard parallel processing system having a blackboard control unit (BCU) including a database module, a trigger module, a scheduler module and a communication module. The BCU also includes a plurality of parallel processors, e.g., knowledge

source processors (KSPs). Skillman et al. are seen to describe a system for employing the KSPs to carry out a specific tasks in accordance with a predefined sequence of sequential processing. As such, Skillman et al. do not describe or suggest the present invention as recited in the independent claims, as now written.

In this regard, the Examiner's attention is respectfully directed to Skillman et al. at Col. 4, line 64 to Col. 5, line 8, and Co. 5, lines 17-29, which recite in pertinent part:

"The interaction between the event driven applications, the exchange of data, and other signals between them to which they respond by beginning their contributory processing, and their scheduling as required for sequential processing of the outputs representing each step of the general processing task are all organized under a blackboard processing system 16 that maps the event driven applications into three possibly heterogeneous parallel computing modules 18(a) through 18(c). Each event driven application can be characterized as a 'knowledge source' because it produces a signal or data required in completing the general processing task. ...

Under blackboard processing system 16, at any given time, various knowledge sources can be running in parallel and accessing a common database, thereby enabling them to use the output from other knowledge sources that have concluded their processing and any other data required to carry out a portion of the general processing task. Blackboard processing system 16 organizes the interaction between knowledge sources 20, ensuring that the appropriate input is applied to each knowledge source when required to enable it to provide its contribution to the general processing task, thereby running each of the event driven applications in an effective and efficient manner." (emphasis added)

At Col. 6, lines 65 to Col. 7, line 30, Skillman et al. describe:

"Carrying the example presented initially in FIG. 1 into the block diagram for the blackboard system shown in FIG. 3, seven knowledge sources 52 are included, distributed between the KSPs 40. Thus, knowledge sources 52(a) through 52(g) run in a combination of parallel and serial sequences, each being assigned to one of the KSPs by BCU 32, consistent with the KSP configuration data set 44. Furthermore, the knowledge sources are initiated when the output from another knowledge source matches one of trigger patterns 48. Accordingly, after knowledge source 52(a) completes the processing of the input signal producing an output, trigger module 46 compares the output to a predefined set of patterns, and if a match is found, triggers scheduler module 42. In response to a trigger signal produced by the trigger module, scheduler module 46 ensures that the output from knowledge source 52(a), which was transferred through communications module 38 to database module 34, is made available to both knowledge source 52(e) within KSP 40(a) and knowledge source 52(b) within KSP 40(b), initiating both of these latter knowledge sources to begin processing.

When knowledge source 52(b) completes its processing task, its output is transferred through communications module 38 to database module 34. A match between this output and a trigger pattern causes trigger module 46 to produce another trigger signal. In response to this trigger signal, scheduler module 42 transfers the data output from knowledge source 52(b) to knowledge sources 52(c) and 52(d). In a similar fashion, each of the knowledge sources implemented within the KSPs are initiated in an orderly sequence, and these processes are distributed between the KSPs to run in parallel when possible, so as to efficiently complete the general processing task.”

As such, Skillman et al. are seen to describe a combination of sequential and parallel processing that invokes a first predefined sequence of instructions in response to a predetermined event. Each processing step of a processing task providing output (via a communication module and a previously stored record in a database) to a next process step of the processing task. That is, the output is made available only by retrieving the output from the database and passing the output to the next processing step.

Additionally, Skillman et al. describe that such output may initiate/spawn execution of a second predefined sequence of instructions, e.g., the BCU detects a trigger condition and initiates execution of another trigger pattern. Such execution is not seen to influence the sequential processing of the first predefined sequence of instructions. In fact, there is no description or suggestion, expressed or implied, that the first executing sequence of sequential instructions would influence the second executing sequence of sequential instructions. As such, Skillman is not seen to describe or suggest selective execution of a next instruction within a sequence of instructions based on dynamic information received and evaluated by a currently executing instruction.

Fischer is seen to describe a “traveling program” for automating data collection among a group of users and, more particularly, for conditionally deciding which of a group of users may participate in a digital signature process for approving various electronic forms.

The Examiner’s attention is respectfully directed to Fischer at Col. 1, lines 48-52, Col. 2, lines 14-22, and Col. 5, line 62 to Col. 6, line 7, which recite, in pertinent part, the object of Fischer’s invention:

“The present invention allows for the travelling program to compute, according to any algorithm whatsoever, the digital material which is to be signed, and also, as needed, the digital material which is to be verified.”

“In addition to being able to sign arbitrary data, the present invention also allows the program to conditionally decide, based on any known criteria, which users should participate in the signature process.”

“The present invention provides a unique mechanism for automating data collection among a group of users. The travelling program may be sent to one user, attach (or detach) relevant data files and move on to the next user. Data or files, collected from one or more users can be deposited with another user, or accumulated for hatched processing as desired. This methodology eliminates the need for individual users to be counted on to transmit all the required data in the required format.

The present invention also efficiently performs electronic document interchange (EDI) in the context of a travelling program which sends itself from [one] user to [] the next within an organization, collecting, editing and approving data.”

Additionally, Fischer describes a predefined “intelligence” (e.g., instructions) for transmitting the traveling program from a first to a next user. For example, Fischer at Col. 5, lines 33-66, recites:

“The engineer using keyboard 4 would access a parts requisition ‘travelling program’ of the type to be described in detail below. The requisition ‘travelling program’ will prompt the engineer to describe the component parts needed. The travelling program includes an instruction sequence which will automatically transmit itself to a next destination, e.g., to a supervisor who has access to terminal B and who is higher up in the organizational structure and possesses the authority to approve the requisition request and digitally sign it. The travelling program may also transmit ancillary information, such as files which may be necessary or useful at future destinations. The supervisor will be prompted to properly digitally sign the request. It is possible that the digital signature reflects not only specific variables values, but also the variable names. Alternatively, the signature may also reflect some aggregate structure which is derived from variables computed within the program, wherein the values may be based on any of many sources, including data read from file, user input, data built into the program, various signer's certificates, or data which is extracted from the user environment (such as the user's ID), etc.

If the request is approved, the requisition form will take a different path in the organization then if it is not approved. The travelling program can have the intelligence to determine, based upon the input from the supervisor at the operating terminal B, where to transmit itself within the organization. The travelling program will also, if desired, load the memory associated with terminal B with the appropriate data relating to the requisition and to attach if desired any files from terminal B that needs to be forwarded elsewhere in the organization.”

Accordingly, Fischer is seen to describe a data structure and mechanism for performing digital signature processing in accordance with a predefined review/approval process (albeit at least partially encoded within the traveling program).

The Examiner suggests that one skilled in the art would be motivated to combine Skillman and Fischer since Skillman's concurrent parallel execution of KSPs lacks a mechanism to generate multiple instances of required data/processing parameters to be dispatched to each KSP.

Firstly, it is respectfully submitted that one skilled in the art of concurrent parallel execution of a sequential set of instructions (Skillman) would not turn to a method for digital signature processing (Fischer) for a mechanism to implement multiple instances of the sequential set of instructions. Secondly, and as described in detail above, the process of initiating and spawning a second execution of a predefined sequence of instructions in response to a trigger within output of a previously executed instruction, provide Skillman the mechanism for implementing multiple sets of instructions in parallel.

Accordingly, it is not seen how the Examiner properly combines Skillman and Fischer, and any attempt to do so can only be made in light of Applicants' disclosure.

Even assuming, arguendo, that the teachings of these references were somehow combined, which is not admitted is suggested absent the Applicants' disclosure, the combined references do not suggest or make obvious the subject matter of the independent claims, as filed and as now written. At best, the proposed combination would describe a system for employing the parallel processing (KSPs) to carry out a predefined sequences of sequential instructions, output of a previously executed first sequence initiating a second sequence (Skillman) wherein the system includes a mechanism for automating data collection among a group of users and conditionally deciding which of a group of users may participate in a digital signature process for approving data collected among the group of users (Fischer).

In particular, the proposed combination of references do not suggest or make obvious the subject matter of the independent claims, as filed and as now written. For example, independent claim 1 recites, in pertinent part:

“1. A data processing system, comprising:
a plurality of event modules ... ;
a plurality of scripts each having a plurality of instructions;
a plurality of processing modules ... ; and
a task module, selectively communicating ..., said task module including code for selecting and instantiating one of said plurality of scripts ... and for executing said instantiated script such that said instantiated script proceeds to a first of said plurality of distributed processing modules for processing a current one of said plurality of instructions;
wherein dynamic information comprises statuses of said distributed processing modules and modifications to said instantiated script including processed data from previously processed ones of said plurality of instructions, and wherein during execution of said instantiated script said task module provides said dynamic information to said instantiated script and incorporates said dynamic information into said currently processing instruction for real-time consideration thereof, and upon completion of said currently processing instruction said task module evaluates said incorporated dynamic information and processed data from said completed instruction and selectively executes said instantiated script such that said instantiated script proceeds to a second of said distributed processing modules for processing a next instruction within said instantiated script.”

Independent claims 33, 34, 60 and 63, as now written, include similar language.

The cited documents are not seen to expressly or implicitly describe or suggest receiving dynamic information within a currently processing instruction, considering such information in real-time and selectively executing an instantiated script (including the currently processing instruction) in response to the dynamic information by proceeding to a second distributed processing modules for processing a next instruction within the instantiated script.

In view of the foregoing, it is respectfully submitted that independent claims 1, 33, 34, 60 and 63 are clearly patentable over the Examiner's proposed combination of Skillman and Fischer. As these independent claims are patentable over the cited documents, the claims that depend from and further limit these independent claims, must also be found to be patentable.

Accordingly, the Examiner is respectfully requested to reconsider and remove the rejection of claims 1-46, 60-61 and 63 under 35 U.S.C. §103(a).

In Section 4 of the Office Action the Examiner rejected claim 62 under 35 U.S.C. §103(a) as being unpatentable over Skillman et al. and Fischer, as applied to claim 60, and further in view of Waclawsky. This rejection is respectfully disagreed with, and is traversed below.

The Examiner states that Waclawsky discloses tracing execution of instructions and provides a mechanism for collecting and analyzing load data from processing modules. Without addressing the characterization of Waclawsky, it is submitted that Waclawsky is not seen to cure the deficiencies cited above with respect to Skillman and Fischer and independent claim 60, for example, where the cited documents do not describe or suggest receiving dynamic information within a currently processing instruction, considering such information in real-time and selectively executing an instantiated script (including the currently processing instruction) in response to the dynamic information by proceeding to a second distributed processing modules for processing a next instruction within the instantiated script.

Since Waclawsky is not seen to cure these deficiencies, claim 62 is deemed patentable over the Examiner's proposed combination of Skillman, Fischer and Waclawsky. Therefore, the Examiner is respectfully requested to reconsider and remove the rejection of claim 62 under 35 U.S.C. §103(a).

As noted above, clarifying amendments are proposed to even further distinguish the independent claims from the cited documents. In view of the foregoing, Applicants submit that independent claims, and the claims that depend therefrom, are patentable over the cited documents.

Accordingly, the Examiner is respectfully requested to reconsider and remove the rejections of all of the pending claims and to allow the application as now presented. If a notice of allowance cannot be issued, it is respectfully requested that the undersigned attorney of record be contacted to resolve any outstanding issues.

Serial No.: 09/030,258
Art Unit: 2151

Atty. Docket No.: 12217-100

Respectfully submitted

DATE: October 4, 2002



Michael K. Kinney, Reg. No. 42,740
Attorney for Applicants

Direct: (203) 498-4411
Email: mkinney@wiggins.com

WIGGIN & DANA LLP
One Century Tower
New Haven, Connecticut 06508
Tel. No. 203-498-4400
Fax. No. 203-782-2889

\\12217\\1\\364772.1

MARKED-UP VERSION OF CLAIMS

1. (Fourth Amendment) A data processing system, comprising:

a plurality of event modules each including code that generates an event data signal representative of a particular event;

a plurality of scripts each having a plurality of instructions;

a plurality of processing modules distributed over said data processing system each including code that provides processed data; and

a task module, selectively communicating with each of said plurality of event modules and said plurality of distributed processing modules, said task module including code for selecting and instantiating one of said plurality of scripts that corresponds to said event data signal and for executing said [instance of said selected] instantiated script such that said [instance of said selected] instantiated script proceeds to a first of said plurality of distributed processing modules for processing a current one of said plurality of instructions;

wherein dynamic information comprises statuses of said distributed processing modules and [said] modifications to said instantiated script including processed data from previously processed ones of said plurality of instructions, and wherein during execution of said [instance of said selected] instantiated script said task module provides said dynamic information to said [instance of said selected] instantiated script and incorporates said dynamic information into said currently processing instruction for real-time consideration thereof, and upon completion of said currently [executing] processing instruction[,], said task module evaluates said incorporated dynamic information and processed data from said completed instruction and selectively executes [, based upon said incorporated dynamic information,] said [instance of said selected] instantiated script such that said [instance of said selected] instantiated script proceeds to a second of said

distributed processing modules for processing a next instruction within said [instance of said selected] instantiated script.

33. (Fourth Amendment) A data processing system, comprising:

a plurality of event modules each including code that generates an event data signal representative of a particular event;

a plurality of scripts each having a plurality of instructions;

a plurality of processing modules distributed over said data processing system each including code for performing data processing functionality to provide processed data;

a task module, selectively communicating with each of said plurality of event modules and said plurality of distributed processing modules, said task module including code for selecting and instantiating one of said plurality of scripts that correspond to said event data signal and, during execution of said [instance of said selected] instantiated script, for providing dynamic information comprising statuses of said distributed processing modules and modifications to said instantiated script including [said] processed data from previously processed ones of said plurality of instructions [to said instance of said selected script] for [incorporation therein] incorporating said dynamic information into said currently processing instruction for real-time consideration thereof and, for selectively executing, based on said incorporated dynamic information and processed data from said completed instruction, said [instance of said selected] instantiated script such that said [instance of said selected] instantiated script proceeds to a first and to at least a second of said distributed processing modules for processing instructions within said [instance of said selected] instantiated script; and

a resource management module communicating with each of said plurality of event modules, said task module and said plurality of distributed processing modules, said resource management module including code for monitoring event data signals generated by said plurality of event modules and not processed by said task module and a number of said plurality of distributed processing modules available for performing particular data processing functionality, and for converting data processing functionality of said plurality of distributed processing modules in response to dynamic information regarding said monitored event data signals and said number of available distributed processing modules to maximize a number of said distributed processing modules processing said event data signals.

34. (Fourth Amendment) A method of data processing comprising the steps of:
generating at least one event data signal at one or more peripheral modules;
mapping said at least one event data signal to a selected script chosen from one or more scripts, each said one or more scripts having one or more instructions;
instantiating said selected script; and
executing, by a task module, said [instance of said selected] instantiated script such that said [instance of said selected] instantiated script proceeds to a first of a plurality of processing modules for processing a current one of said one or more instructions of said [instance of said selected] instantiated script;

wherein dynamic information comprises statuses of said plurality of processing modules and modifications to said instantiated script including processed data [provided by said plurality of processing modules] from previously processed ones of said one or more instructions, and wherein during execution of said [instance of said selected] instantiated script said task module provides

said dynamic information to said [instance of said selected] instantiated script and incorporates said dynamic information into said currently processing ones of said one or more instructions for real-time consideration thereof [for incorporation therein], and upon completion of said currently [executing] processing instruction, said task module evaluates said incorporated dynamic information and processed data from said completed instruction and selectively executes [, based upon said incorporated dynamic information,] said [instance of said selected] instantiated script such that said [instance of said selected] instantiated script proceeds to a second of said plurality of processing modules for processing a next instruction within said [instance of said selected] instantiated script.

60. (Second Amendment) In a data processing system, a method for responding to event data, comprising:

receiving event data from a requesting one of a plurality of event modules;

mapping the event data to a selected one of a plurality of scripts, the plurality of scripts including instructions for responding to event data;

instantiating said selected script;

executing, by a task module, the [instance of the selected] instantiated script such that the [instance of the selected] instantiated script proceeds to a first of a plurality of processing modules for processing of a current one of the instructions of the [instance of the selected] instantiated script;

wherein dynamic information comprises statuses of the plurality of processing modules and modifications to the instantiated script including processed data [provided by the processing modules] from previously processed ones of the instructions, and wherein during the execution

of the [instance of the selected] instantiated script the task module provides the dynamic information to the [instance of the selected] instantiated script [for incorporation therein] and incorporates the dynamic information into the currently processing instruction for real-time consideration thereof, and upon completion of the currently [executing] processing instruction the task module evaluates the incorporated dynamic information and processed data from the completed instruction and selectively executes [, based on the incorporated information,] the [instance of the selected] instantiated script such that the [instance of the selected] instantiated script proceeds to a second of the plurality of processing modules for processing a next instruction within the [instance of the selected] instantiated script;

building a response profile including results generated during execution of the [instance of the selected] instantiated script; and

wherein when the instructions within the [instance of the selected] instantiated script are completed, transmitting the response profile to the requesting one of the plurality of event modules.

63. (Second Amendment) A data processing system, comprising:

a plurality of event modules each including code that generates a first event data signal representative of a first event;

a plurality of scripts each having a plurality of instructions;

a plurality of processing modules each including code that provides processed data, a subset of said plurality of processing modules having code that selectively generates a second event data signal representative of a second event; and

a task module, selectively communicating with each of said plurality of event modules and said plurality of processing modules, said task module including code for selecting and instantiating ones of said plurality of scripts that corresponds to said first and second event data signals, and for executing said [instances of said selected] instantiated scripts such that said [instances of said selected] instantiated scripts proceed to a first of said plurality of processing modules for processing a current one of said plurality of instructions within each of said instances;

wherein dynamic information comprises statuses of said plurality of processing modules and [said] modifications to said instantiated scripts including processed data from previously processed instructions, and wherein during execution of said [instances of said selected] instantiated scripts, said task module provides said dynamic information to said [instances of said selected] instantiated scripts [for incorporation therein] and incorporates said dynamic information into said currently processing instructions for real-time consideration thereof, and upon completion of said currently [executing] processing instructions, said task module evaluates said incorporated dynamic information and processed data from said completed instructions and selectively executes [, based on said incorporated dynamic information,] said [instances of said selected] instantiated scripts such that [instances of said selected] instantiated scripts proceed to a second of said processing modules for processing of a next instruction within said [instances of said selected] instantiated scripts.